DATA · TAUGHT WITH POSTGRESQL

Databases

SQL, schema design, transactions and the query planner. You write queries that are correct before they are fast, then make them fast by understanding what Postgres is actually doing.

Foundational | 5 phases | 23 sessions | 35 contact hours

Syllabus

5 phases, 23 sessions. Each phase ends in something you have built.

1 Getting Correct Answers

Get a correct answer out of a database you did not design.

6 sessions
$58 alone

1.1 The Relational Model

Tables, rows, keys, and why the order of rows means nothing.

FREE PREVIEW

1.2 Your First Queries

SELECT, WHERE, ORDER BY, LIMIT, and reading the error message instead of guessing.

1.3 Joins and Join Predicates

INNER, LEFT, and the missing ON clause that quietly returns every pair of rows.

1.4 Duplicates, Grouping, and Aggregates

Why your COUNT is too high: join fan-out, GROUP BY grain, and DISTINCT as a symptom.

1.5 NULL Is Not a Value

Three-valued logic, IS DISTINCT FROM, and why NOT IN with a NULL returns nothing at all.

1.6 Subqueries and EXISTS

Scalar and correlated subqueries, EXISTS as the safe rewrite, UNION and EXCEPT.

By the end: **A report answering 20 questions about a real dataset, with every row count justified.**

2 Designing a Schema

Design tables where the wrong data cannot be stored in the first place.

5 sessions
$58 alone

2.1 From Requirements to Tables
Entities, relationships, cardinality, and the join table every many-to-many needs.

2.2 Normalization
First through third normal form, and the update anomaly each one removes.

2.3 Choosing Types
text, numeric, timestamptz, enums, and the types Postgres itself tells you to avoid.

2.4 Keys and Constraints
Identity columns, uuidv7, NOT NULL, CHECK, foreign keys, and what NOT ENFORCED is for.

2.5 Migrations
Versioned schema changes, reversibility, and the ALTER TABLE that locks the table for an hour.

By the end: **A schema built by versioned migrations that rejects bad rows on insert.**

3 How Postgres Actually Works

Explain what Postgres does between BEGIN and the bytes on disk.

4 sessions
$58 alone

3.1 Pages, Heaps, and TOAST
How a row is laid out on disk and what happens to values too big to fit in a page.

3.2 Transactions and Isolation
BEGIN, COMMIT, and what read committed, repeatable read, and serializable each guarantee.

3.3 Locks and Deadlocks
Row and table locks, the queue behind them, and reading pg_locks when everything stops.

3.4 MVCC and Vacuum
Row versions, dead tuples, bloat, autovacuum, and freezing before the wraparound warning.

By the end: **A concurrency anomaly reproduced in two psql sessions, plus the isolation level that stops it.**

4 Making Queries Fast

Explain why a query is slow before adding an index.

4 sessions
$58 alone

4.1 How an Index Works

B-tree lookups end to end, and where GIN, GiST, and BRIN beat a B-tree.

4.2 Reading a Query Plan

EXPLAIN ANALYZE, estimated versus actual rows, and the buffer counts now printed by default.

4.3 Indexing Strategy

Column order, composite, partial, covering and expression indexes, and skip scan in Postgres 18.

4.4 Statistics and Tuning

ANALYZE, why the planner guessed wrong, pg_stat_statements, work_mem, and connection pooling.

By the end: **One slow query made ten times faster, with the plans from before and after.**

5 Running It For Real

Keep the database available through failure, growth, and your own mistakes.

4 sessions
$58 alone

5.1 Backup and Recovery

Logical dumps, physical base backups, WAL archiving, and restoring to a chosen timestamp.

5.2 Replication and Failover

Streaming replicas, replication lag, read routing, and what a failover actually loses.

5.3 Partitioning and Upgrades

Declarative range partitioning, and a major version upgrade that keeps its optimizer statistics.

5.4 Postgres Beyond Relational

jsonb, full-text search, pgvector similarity search, and when a separate store is worth the cost.

By the end: **A replicated cluster restored to a chosen timestamp after you break it on purpose.**

Tools you will use

PostgreSQL 18 | psql | DBeaver | Docker | Flyway | pgbench | pg_stat_statements | pgvector

What you will build

1. **Schema Rescue**
   Turn a 40-column spreadsheet into tables that reject bad rows
2. **The 10x Query**
   EXPLAIN first, index second, on a million rows
3. **Restore Drill**
   Break it on purpose, recover to a timestamp

Registration

Phase 1: Getting Correct Answers	$58
Phase 2: Designing a Schema	$58
Phase 3: How Postgres Actually Works	$58
Phase 4: Making Queries Fast	$58
Phase 5: Running It For Real	$58
Whole track, all 5 phases	**$240**

Buying every phase separately costs $290, so the whole track saves $50.

No payment is taken online. Registering creates an order and payment instructions are sent with it. Access opens once payment is confirmed.